Kapasiteettirajoitettujen ajoneuvojen reititysongelma

LuK-tutkielma
Huhtikuu 2025

MATEMATIIKAN JA TILASTOTIETEEN LAITOS

TURUN YLIOPISTO
Matematiikan ja tilastotieteen laitos

Henrik Kauppi: Kapasiteettirajoitettujen ajoneuvojen reititysongelma
LuK-tutkielma
Matematiikka
Huhtikuu 2025

Tutkielmassa esitellään yleisesti kapasiteettirajoitettujen ajoneuvojen reititysongelma (CVRP). Tämä on kauppamatkustajan ongelman laajennus ja yksi tutkituimmista matemaattisen optimoinnin ongelmista. Ongelma muodostuu kapasiteettirajoitetuista ajoneuvoista, joiden reitit asiakkaille optimoidaan esimerkiksi niiden matkan tai kustannusten minimoimiseksi. Kapasiteettirajoitukset tarkoittavat mitä vain kuormallista rajoitetta. Näitä voivat olla esimerkiksi tilataksin matkustajamäärä, toimitettavien pakettien määrä tai kokonaistilavuus. Kapasiteettirajoitus voi koskea toimitusten enimmäiskuorman lisäksi myös sitä, kuinka paljon tavaraa voidaan noutaa asiakkailta. Oikean maailman ongelmiin liittyy usein kapasiteettirajoitusten lisäksi muita tehtäväkohtaisia rajoituksia, kuten ajoneuvojen määrä tai ajomatkan enimmäispituus. Ongelmalle on monta eri matemaattista mallinnustapaa. Tutkielmassa esitellään lineaarinen kokonaislukuohjelmointimalli, kolmen indeksin ajoneuvovirtausmalli ja ositusmalli.

CVRPt ovat NP-vaikeita ongelmia, eli niiden täsmällisten ratkaisujen laskeminen on vaikeaa varsinkin suurilla asiakasmäärillä. Ongelman monimutkaisuuden takia suuri osa tutkimuksista on keskittynyt erilaisten tehokkaiden epätäsmällisten menetelmien kehittämiseen. Menetelmien tehokkuus on usein tapauskohtaista, ja niitä voidaan arvioida esimerkiksi vertailemalla tarvittavaa laskennallista tehoa tai ratkaisujen optimaalisuutta. Tutkielmassa tarkastellaan yleisesti eri menetelmätyyppejä ja tarkemmin yhtä heuristista asiakkaiden ryhmittelyyn perustuvaa menetelmää: Ryhmittele ensin, reititä sitten (CFRS). Menetelmä perustuu nimensä mukaisesti siihen, että ensin ryhmitellään joukko asiakkaita eri ryppäisiin, minkä jälkeen optimoidaan ryppäiden asiakkaille ajoneuvojen toimitusreitit. Ryhmittelyn tarkoituksena on vähentää tarvittavien muuttujien määrää CVRP:n ratkaisemiseksi. Menetelmä on osoittautunut erityisen tehokkaaksi laskenallisten kustannusten ja ratkaisujen optimaalisuuserojen testeissä pienissä 32–82 asiakaspisteen tapauksissa.

Asiasanat: kapasiteettirajoitettujen ajoneuvojen reititysongelma (CVRP), ryhmittele ensin, reititä sitten -menetelmä (CFRS).

Sisällys

1 Johdanto

Matemaattisessa optimoinnissa etsitään parasta mahdollista ratkaisua tarkastelun kohteena olevalle ongelmalle. Ennen kuin ongelma voidaan ratkaista, se on esitettävä matemaattisesti optimointitehtävänä. Tätä varten on määrättävä päätösmuuttujat, kohdefunktio ja rajoitteet, joiden ominaisuudet määrittävät yhdessä optimointitehtävän.

Päätösmuuttujat ovat ne tuntemattomat arvot, joita optimoidaan ja joiden avulla määritetään optimiratkaisu. Päätösmuuttujat voidaan jakaa jatkuviin, diskreetteihin ja binäärisiin muuttujiin. Jatkuvat päätösmuuttujat ovat reaalilukuja, esimerkiksi tuotantomäärä. Diskreetit päätösmuuttujat kuvaavat muuttujia kokonaisluvuilla, esimerkiksi ajoneuvojen lukumäärää. Binääriset päätösmuuttujat taas usein kuvaavat kyllä- tai ei-päätöksiä.

Kohdefunktio määrittää, mitä asiaa optimointitehtävässä optimoidaan. Kohdefunktiot voidaan jakaa minimointitehtäviin ja maksimointitehtäviin. Minimointitehtävissä tavoitteena on minimoida kohdefunktioita. Tällaisia ongelmia voivat olla esimerkiksi kuljetuskustannusten minimointi tai ajoreitin ajamiseen käytetyn ajan minimointi. Maksimointitehtävissä vastaavasti tavoitteena on maksimoida kohdefunktioita. Käytännön esimerkkejä maksimointiongelmille ovat esimerkiksi kokonaisvoiton maksimointi tai reitin tehokkuuden maksimointi.

Rajoitteet asettavat ne ehdot, joiden puitteissa optimointi tapahtuu. Ne voivat olla esimerkiksi resurssien määrärajoja tai teknisiä vaatimuksia. Rajoitteet voidaan jakaa kahteen kategoriaan, luonnollisiin rajoitteisiin ja varsinaisiin rajoitteisiin. Luonnollinen rajoite on sellainen rajoitus, joka seuraa suoraan ongelman fyysisistä, loogisista tai käytännöllisistä ominaisuuksista. Se ei ole keinotekoisesti lisätty, vaan välttämätön ongelman määrittelyn kannalta. Esimerkiksi työvuoroja suunniteltaessa tällainen rajoite voisi olla se, että työntekijä ei voi työskennellä useammassa kuin yhdessä vuorossa samanaikaisesti. Varsinainen rajoite taas tarkoittaa rajoitusta, joka ei ole vain luonnollinen seuraus ongelman asetelmasta vaan asetetaan nimenomaan ratkaisun ohjaamiseksi. Tällainen rajoite voisi olla, että jokaiselle vuorolle on vähintään yksi mutta enintään kolme työntekijää.

Optimointitehtävät voidaan jakaa lineaarisiin ja epälineaarisiin tehtäviin. Lineaarisessa optimoinnissa kohdefunktio ja rajoitteet ovat lineaarisia. Sen sijaan tehtävät ovat epälineaarisia, jos kohdefunktio f tai mikään epäyhtälörajoitteista $c_i(x) \leq 0, \quad i = 1, 2, \ldots, m$ tai yhtälörajoitteista $d_j(x) = 0, \quad j = 1, 2, \ldots, n$ on epälineaarinen. Epälineaarisia ovat ongelmat, joita ei voi muotoilla lineaarisina funktioina ilman, että menettää jonkin tärkeän piirteen oikean maailman epälineaarisuuden vuoksi. Esimerkki tällaisesta on rahoituksessa sijoitussalkun riski, joka mitataan salkun tuoton varianssina. Se on epälineaarinen funktio kuhunkin arvopaperiin sijoitetusta määrästä. [6]

Yksi tunnetuimpia ja tutkituimpia matemaattisen optimoinnin ongelmia on ajoneuvojen reititysongelma (englanniksi *Vehicle Routing Problem* eli *VRP*). Ajoneuvojen reititysongelma on laajennettu ongelma yksinkertaisemmasta kauppamatkustajaongelmasta (englanniksi *Travelling Salesman Problem* eli *TSP*). Kauppamatkustajaongelmassa etsitään lyhintä reittiä kaupunkien ja varastopisteiden välille. Ongelmassa määrätään "kauppamatkustaja"käymään jokaisessa kaupungissa ja va-

rastopisteessä täsmälleen kerran ja palaamaan lähtöpisteeseen. VRP:ssä taas pyritään suunnittelemaan mahdollisimman tehokkaat reitit ajoneuvoille, jotka kuljettavat lähetyksiä eri kohteisiin. Tavoitteena on vähentää resurssien, kuten kustannusten, polttoaineen ja ajan, käyttöä ja samalla parantaa toiminnan tehokkuutta. VRP ei kuitenkaan rajoitu vain lyhimmän reitin löytämiseen, vaan siinä on huomioitava myös käytännön rajoitteet, kuten ajoneuvojen kapasiteetti, toimitusaikataulut ja muut logistiikkaan vaikuttavat tekijät. [14]

Tämä ongelma on ollut operatiivisen tutkimusyhteisön keskiössä yli 50 vuoden ajan sen taloudellisen merkityksen ja matemaattisten haasteiden vuoksi. Kauppamatkustajaongelma voidaan nykyään ratkaista tuhansille tai jopa kymmenille tuhansille solmuille (nodes). Solmut tarkoittavat sijainteja, joissa kauppamatkustaja vierailee, kuten kaupunkeja tai asiakaspisteitä. VRP on kuitenkin huomattavasti monimutkaisempi. Yksinkertaisestakin ajoneuvojen reititysongelmasta tulee haastava ratkaista, kun tarkkoja algoritmeja sovelletaan yli sadan asiakkaan tapauksissa ja ratkaisussa otetaan huomioon ajoneuvon kapasiteetti. Tällaista ajoneuvojen reititysongelmaa kutsutaan kapasiteettirajoitettujen ajoneuvojen reititysongelmaksi (englanniksi *Capacitated Vehicle Routing Problem* eli *CVRP*). [9]

CVRPt ovat NP-vaikeita ongelmia eli niiden täsmällisten ratkaisujen laskeminen on pitkäkestoista varsinkin suurilla asiakasmäärillä [7]. Ongelman monimutkaisuuden takia suuri osa tutkimuksista on keskittynyt erilaisten tehokkaiden heuristisen menetelmien kehittämiseen. Nämä käyttävät käytännön sääntöjä tai alakohtaista tietoa löytääkseen likimääräisiä ratkaisuja, jotka ovat riittävän lähellä optimaalista ratkaisua, jolloin ne eivät vaadi yhtä paljon laskennallista tehokkuutta [10].

Gilbert Laporte, Paolo Toth ja Daniel Vigo kirjoittivat VRP:n historiasta teoksessaan [9]. Dantzig ja Ramser (1959) esittelivät ensimmäisen CVRP:n muotoilun ja ratkaisivat sen yksinkertaisella heuristiikalla. Seuraavina vuosina kehitettiin useita heuristiikkoja, kuten Clarke ja Wrightin (1964) säästöheuristiikka, joka on edelleen laajasti käytössä sen nopeuden ja kohtuullisen tarkkuuden ansiosta.

VRP:n tarkkojen ratkaisumenetelmien kehitys alkoi vuonna 1981, kun Christofides, Mingozzi ja Toth esittivät ensimmäiset matemaattiset muotoilut ja dynaamiseen ohjelmointiin perustuvan ratkaisualgoritmin. Laporte, Desrochers ja Nobert (1984) puolestaan kehittivät ensimmäisen leikkaustasomenetelmän, joka vaikutti moniin myöhempiin tarkkoihin algoritmeihin.

Sittemmin on kehitetty useita tarkkoja menetelmiä, kuten haarautus ja leikkaus (branch-and-cut) ja joukkojakomenetelmät (set partitioning), joiden avulla VRP voidaan ratkaista tehokkaammin. Merkittäviä edistysaskeleita on nähty esimerkiksi Fukasawan et al. (2006) ja Baldaccin et al. (2008) tutkimuksissa.

VRP:iden tutkiminen koki merkittävän murroksen 1990-luvulla, kun metaheuristiset menetelmät, kuten tabu-haku, mukautuva laajan naapuruston haku (englanniksi *Adaptive Large Neighborhood Search* eli *ALNS* ja geneettisten algoritmien hybridit alkoivat yleistyä. Viime vuosina menetelmät ovat kehittyneet kokonaisvaltaisemmiksi ja kykenevät ratkaisemaan useita VRP-muunnelmia.

Viimeaikaisessa tutkimuksissa VRP-malleja on laajennettu huomioimaan aikaikkunat, synkronointi ja stokastiset rajoitteet. Stokastiset rajoitteet tarkoittavat rajoitteita, jotka esiintyvät optimointitehtävissä epävarmuustekijöinä, joissa rajoitteiden parametrit tai muuttujat eivät ole kiinteitä arvoja vaan satunnaismuuttujia.

Tämä tarkoittaa, että rajoitteet eivät aina päde deterministisesti, vaan ne toteutuvat tietyllä todennäköisyydellä. Lisäksi grafiikkasuoritinpohjainen rinnakkaislaskenta on noussut tärkeäksi työkaluksi, jolla voidaan ratkaista suuria VRP-tapauksia reaaliajassa.

Tämä tutkielma keskittyy erityisesti kapasiteettirajoitettujen ajoneuvojen reititysongelmaan (CVRP), joka on yksi käytännössä tärkeimmistä VRP-muunnelmista. CVRP:tä sovelletaan muun muassa logistiikassa, jakelukuljetuksissa ja toimitusketjuissa, kun ajoneuvojen reitit on suunniteltava tehokkaasti, huomioiden niiden rajallinen kapasiteetti.

Tutkielmassa tarkastellaan, kuinka CVRP voidaan mallintaa matemaattisesti ja ratkaista erilaisilla algoritmeilla. Luku 2 käsittelee CVRP:n mallinnusta eri mallinnustavoilla: lineaarinen kokonaislukuohjelmointimalli, kolmen indeksin ajoneuvovirtausmalli ja ositusmalli. Luku 3 käsittelee yleisesti CVRP:n ratkaisumenetelmiä. Siinä esitellään myös ryhmittelyyn perustuva heuristinen ratkaisumenetelmä: ryhmittele ensin, reititä sitten (CFRS). Lopuksi luvussa 4 tehdään yhteenveto työn keskeisistä sisällöistä.

2 Reitittämisen mallinnus

Kapasiteettirajoitettujen ajoneuvojen reititysongelman tavoitteena on etsiä optimaalisin reitti toimituspisteiden välille ajoneuvoille, joilla on rajallinen kuorman kapasiteetti. Kapasiteettirajoitus voi periaatteessa tarkoittaa mitä vain kuormallista rajoitetta. Näitä voivat olla esimerkiksi tilataksin matkustajamäärä, toimitettavien pakettien määrä, pakettien kokonaispaino tai kokonaistilavuus. Kapasiteettirajoitus voi koskea toimitusten enimmäiskuorman lisäksi myös sitä, kuinka paljon tavaraa voidaan noutaa asiakkailta. Esimerkiksi jätekeräyksessä roska-auton kapasiteettirajoitus määrittää kuinka suuren jätemäärän ajoneuvo voi ottaa kyytiin ennen tyhjennystä.

Ajoneuvojen on tarkoitus lähteä varastolta, käydä läpi ajoneuvolle määrätyt asiakkaat, ja palata sen jälkeen takaisin varastolle. Kun asiakkaiden osoitteet ja pakettien koko on tiedossa, voidaan jokaiselle ajoneuvolle määrätä yksilöllinen optimaalinen reitti, jolloin esimerkiksi kustannuksia, käytettyjen ajoneuvojen määrää tai kuljetuksiin kuluvaa aikaa saadaan minimoitua. Yleensä kuitenkin kapasiteetin lisäksi ongelmiin sovelletaan muitakin rajoituksia, kuten aikarajoitteet ja maksimi toimintamatka.

Tehtävän mallinnus toteutetaan rakentamalla matemaattinen esitystapa, joka kuvaa ongelman eri osa-alueet. Näihin kuuluu esimerkiksi asiakkaiden osoitteiden sijainti ja niiden välinen etäisyys sekä etäisyys varastoon. Lisäksi huomioidaan ajoneuvojen tilavuusrajoitukset, jotka vaikuttavat kuljetussuunnitteluun. Seuraavaksi määrätään päätösmuuttujat, jotka kuvaavat mitkä ajoneuvot kulkevat kunkin toimipisteen kautta. Kohdefunktio taas luodaan tehtävän tavoitteen mukaan esimerkiksi minimoimaan kustannuksia tai ajoneuvojen kulkemaa kokonaismatkaa, mihin vaikuttavat yksittäisten ajoneuvojen reittipituudet ja polttoainekulut. Lopuksi malliin lisätään rajoitteet, kuten se, että ajoneuvojen kapasiteetit eivät saa ylittyä, jokaisella toimipisteellä käydään vain kerran ja jokaisen ajoneuvon on aloitettava ja lopetettava reittinsä varastolla.

CVRP:n ratkaisemiseksi on kehitetty useita matemaattisia malleja, jotka eroavat toisistaan tehokkuuden, tarkkuuden ja laskennallisen monimutkaisuuden suhteen. Tässä luvussa esitellään kolme mallia:

- Lineaarinen kokonaislukuohjelmointimalli
- Kolmen indeksin ajoneuvovirtausmalli
- Ositusmalli

Seuraavaksi käsitellään jokainen malli erikseen.

2.1 Lineaarinen kokonaislukuohjelmointimalli

Yksi tapa mallintaa kapasiteettirajoitettujen ajoneuvojen reititysongelmaa on muotoilla se lineaariseksi kokonaislukuohjelmointimalliksi (englanniksi *Linear Integer Programming Model*). Seuraavaksi esitellään kyseinen malli lähteen [1] mukaisesti siten, että reittien kokonaismatkaa minimoidaan.

Vastaavasti kuin TSP:ssä solmut (*node*) tarkoittavat verkoston pisteitä, joissa ajoneuvot voivat käydä. CVRP-mallissa solmut edustavat asiakkaita, joille toimituksia tehdään, tai varastoa, josta ajoneuvot lähtevät ja johon ne palaavat. Kaari (*arc*) puolestaan yhdistää kaksi solmua ja edustaa mahdollista reittiä niiden välillä. Esimerkiksi, jos ajoneuvo kulkee asiakkaalta 1 asiakkaalle 2, se käyttää kaarta $(1, 2)$. Binäärimuuttuja x_{ijk} on päätösmuuttuja, jota käytetään kuvaamaan kuuluuko tietty reitti ajoneuvon optimaaliseen reittiin seuraavasti

$x_{ijk} = 1$, jos ajoneuvo k kulkee solmusta i solmuun j ja
$x_{ijk} = 0$, jos kyseistä reittiä ei käytetä $\quad \forall k \in \{1, \ldots, p\}, \quad i, j \in \{1, \ldots, n\}.$

Myös siirtyminen solmusta itseensä on kielletty eli

$$x_{iik} = 0 \quad \forall k \in \{1, \ldots, p\}, \quad i, j \in \{1, \ldots, n\}.$$

Parametri d_{ij} kuvaa etäisyyttä solmusta i solmuun j. Verkossa on yhteensä n solmua, joista yksi on varasto. Käytössä on p ajoneuvoa. Kohdefunktio voidaan esittää siis seuraavasti

$$\min \sum_{k=1}^{p} \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_{ijk}.$$

Jokaisessa solmussa tulee käydä yksittäisen ajoneuvon ja sieltä tulee poistua täsmälleen kerran samalla ajoneuvolla, lukuunottamatta varastoa. Varastosta tulee lähteä ja sinne tulee palata täsmälleen kerran jokaisella ajoneuvolla. Parametri q_i kuvaa kunkin asiakkaan kysyntää eli kuinka paljon kapasiteetista menee asiakkaalle ja Q taas on ajoneuvojen kapasiteetti. Ajoneuvon k palvelemien asiakkaiden kysyntöjen summan ei tule ylittää kyseisen ajoneuvon kapasiteettia. Nämä rajoitteet voidaan

esittää seuraavasti

$$\texttt{s.t.} \quad \sum_{i=1}^{n} x_{ijk} = \sum_{i=1}^{n} x_{jik}, \quad \forall j \in \{1, \ldots, n\}, \quad k \in \{1, \ldots, p\} \tag{1}$$

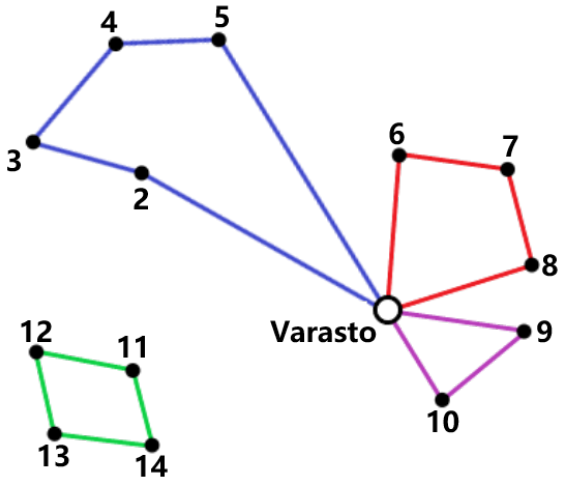
$$\sum_{k=1}^{p} \sum_{i=1}^{n} x_{ijk} = 1, \quad \forall j \in \{2, \ldots, n\} \tag{2}$$

$$\sum_{j=2}^{n} x_{1jk} = 1, \quad \forall k \in \{1, \ldots, p\} \tag{3}$$

$$\sum_{i=1}^{n} \sum_{j=2}^{n} q_j x_{ijk} \leq Q, \quad \forall k \in \{1, \ldots, p\} \tag{4}$$

Rajoite (1) takaa, että ajoneuvo poistuu solmusta, johon se saapuu. Se varmistaa, että ajoneuvon saapumisten määrä solmuun on yhtä suuri kuin sen lähtöjen määrä kyseisestä solmusta. Yhdessä ensimmäisen rajoitteen kanssa rajoite (2) varmistaa, että jokaisessa solmussa käydään vain kerran ja sieltä myös poistutaan samalla ajoneuvolla. Rajoite (3) pitää huolen siitä, että jokainen ajoneuvo lähtee varastolta, ja yhdessä rajoitteen (1) kanssa varmistaa, että jokainen ajoneuvo myös palaa takaisin varastolle. Rajoite (4) katsoo, että ajoneuvon kapasiteettirajoitusta ei ylitetä. On syytä huomioida, että tässä mallissa kaikilla autoilla on sama maksimikapasiteetti.

Mallissa on kuitenkin toistaiseksi yksi oikean maailman ongelma. Rajoitteissa ei oteta huomioon sitä, että optimiratkaisu saattaa sisältää kuvan 1 havainnollistavan alikierroksen (englanniksi *subtour*). Alikierrosten poistamiseen on kehitetty useita tapoja. Yksi tavoista on silmukan ehtojen (englanniksi *loop conditions*) lisääminen rajoitteisiin. George Dantzig, Ray Fulkerson ja Selmer Johnson julkaisivat menetelmän kauppamatkustajaongelman ratkaisemiseksi. Ratkaisussa esiteltiin silmukan ehdot, jotka tunnetaan nykypäivänä alikierrosten rajoitteina (englanniksi *subtour constraints*).[5]



Kuva 1: Havainnollistus alikierroksesta, joka kulkee solmujen 11, 12, 13 ja 14 läpi.

Seuraavaksi esitellään lähteen [2] mukaan Dantzig-Fulkerson-Johnson malli (DFJ).

Malli poistaa alikierrokset käyttämällä osajoukkoja rajoittamaan ei-toivottuja reittejä. Kaikki solmut muodostavat yhdessä joukon

$$V = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 \text{ ja } 14.$$

Alikierroksen solmut $11, 12, 13$ ja 14 muodostavat joukon S, joka on joukon V osajoukko. Osajoukossa S on neljä solmua ja kaarta, minkä takia se on alikierros. Kaarien määrän tulisi olla pienempi kuin neljä eli yksi kaarista tulisi poistaa. Poistetaan kaari solmujen 11 ja 14 väliltä. Nyt solmuista 11 ja 14 täytyy lähteä kaaret solmuihin, jotka eivät ole osajoukossa, eli osajoukosta ulkopuolisiin solmuihin johtavia kaaria on oltava vähintään kaksi. Tämän ehdon täytyy toteutua jokaisen osajoukon kohdalla, jotta kaikki alikierrokset saadaan poistettua. Osajoukot, jotka sisältävät varaston, tai osajoukot, joissa on nolla tai yksi solmua, eivät voi olla alikierroksia. Ehdon tulee koskettaa kaikkia mahdollisia joukon V osajoukkoja, joissa on kaksi solmua ja joissa ei ole varastoa. Tämä ehto voidaan esittää seuraavasti

$$\sum_{i \in S, j \notin S} x_{ijk} \geq 2 \quad S \subset V \setminus \{1\}, \quad 2 \leq |S| \leq n - 2.$$

Tämä lineaarinen kokonaislukuohjelmointimalli tarjoaa CVRP:n ratkaisemiseen täsmällisen lähestymistavan. Malli tuottaa optimaalisia ratkaisuja, mikä tekee siitä erityisen hyödyllisen pienille ja keskikokoisille ongelmille.

2.2 Kolmen indeksin ajoneuvovirtausmalli

Kolmen indeksin ajoneuvovirtausmalli (englanniksi *A three-index vehicle flow formulation*) on Fisherin ja Jaikumarin (1978, 1981) kehittämä CVRP:n muotoilu, jossa ei ole vaatimuksena sitä, että kaikilla ajoneuvoilla on sama kapasiteetti. Kapasiteettirajoitusten lisäksi mallin rajoituksiin on lisätty aikaikkunat ilman pysähdysaikoja. Tällaisessa mallissa käytetään muuttujia kuvaamaan ajoneuvon kulkua kaaren (i, j) kautta. Kolmen indeksin mallissa muuttuja x_{ijk} osoittaa, kulkeeko ajoneuvo k kaaren (i, j) kautta. [8] Seuraavaksi esitellään kyseinen malli lähteen [8] mukaisesti. Olkoot

D_k	ajoneuvon k kapasiteetti,
$[a_i, b_i]$	solmun i aikaikkuna,
t_{ij}	matkaan kuluva aika solmusta i solmuun j,
$x_{ijk} (i \neq j)$	binäärimuuttuja, joka saa arvon 1, jos ajoneuvo k kulkee kaaren (i, j) kautta,
y_{ik}	binäärimuuttuja, joka saa arvon 1, jos ajoneuvo k käy solmussa i,
t_i	ajoneuvon saapumisaika solmussa i,
T	suuri vakio.

Malli muotoillaan seuraavasti

$$\min \quad \sum_{k=1}^{p} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} c_{ij} x_{ijk} \tag{5}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} d_i y_{ik} \leq D_k \quad (k = 1, \ldots, p) \tag{6}$$

$$\sum_{k=1}^{p} y_{ik} = \begin{cases} p & (i = 1) \\ 1 & (i = 2, \ldots, n) \end{cases} \tag{7}$$

$$\sum_{i=1}^{n} x_{ijk} = y_{jk} \quad (j = 1, \ldots, n; \quad k = 1, \ldots, p) \tag{8}$$

$$\sum_{j=1}^{n} x_{ijk} = y_{jk} \quad (i = 1, \ldots, n; \quad k = 1, \ldots, p) \tag{9}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1 \quad (S \subset V; \quad |S| \geq 2; \quad k = 1, \ldots, p) \tag{10}$$

$$t_j \geq t_i + t_{ij} - (1 - x_{ijk})T \quad (i, j = 1, \ldots, n; \quad k = 1, \ldots, p) \tag{11}$$

$$t_j \leq t_i + t_{ij} + (1 - x_{ijk})T \quad (i, j = 1, \ldots, n; \quad k = 1, \ldots, p) \tag{12}$$

$$a_i \leq t_i \leq b_i \quad (i = 2, \ldots, n) \tag{13}$$

$$x_{ijk} \in \{0, 1\} \quad (i, j = 1, \ldots, n; \quad k = 1, \ldots, p) \tag{14}$$

$$y_{ik} \in \{0, 1\} \quad (i = 1, \ldots, n; \quad k = 1, \ldots, p) \tag{15}$$

jossa rajoite (6) on ajoneuvojen kapasiteetteja koskeva rajoitus. Rajoite (7) on ajoneuvojen määritys lähtöpisteelle ja asiakkaille. Lähtösolmu $i = 1$ eli varasto, palvelee kaikkia ajoneuvoja, joten sillä on arvo m. Jokainen asiakas $i \neq 1$ palvellaan tasan yhdellä ajoneuvolla k eli $y_{ik} = 1$. Rajoite (8) varmistaa, että sisään tulevien kaarien määrä on sama kuin asiakasta palvelevien ajoneuvojen määrä. Jos asiakas j kuuluu ajoneuvon k reittiin, sen täytyy olla myös jossakin kaarella kyseisen ajoneuvon reitillä. Rajoite (9) katsoo, että uloslähtevien kaarien määrä on sama kuin asiakasta palvelevien ajoneuvojen määrä. Jos asiakas j kuuluu ajoneuvon k reittiin, sen täytyy myös lähteä kyseiseltä asiakkaalta toiselle asiakkaalle tai varastolle. Rajoitteella (10) poistetaan alikierrokset. Rajoite (11) koskee ajoneuvojen saapumisaikaa. Tämä varmistaa, että asiakas j voi vastaanottaa tilauksen vasta, kun ajoneuvo k on ensin saapunut asiakkaalle i. T on suuri vakio, joka tekee ehdosta merkityksettömän silloin, kun $x_{ijk} = 0$ eli kun ajoneuvo ei kulje kyseistä kaarta. Rajoite (12) on aikaikkunarajoitus. Tämä varmistaa, että asiakas j saa toimituksen aikaisintaan, kun $t_i + t_{ij}$, mutta ei liian aikaisin. Tämä rajoite on erityisen tärkeä aikaikkunoita sisältävissä VRP-muunnelmissa. Rajoite (13) varmistaa, että toimitukset tapahtuvat aikataulun mukaisesti. Jokaisen asiakkaan i palveluajan täytyy olla sen omassa aikaikkunassa $[a_i, b_i]$. Rajoitteet (14) ja (15) ovat binäärimuuttujia. (14) määrittää, käyttääkö ajoneuvo k kaarta (i, j) vai ei, ja (15) määrittää palveleeko ajoneuvo k asiakasta i.

Mallissa on etuna se, että se tuottaa toteuttamiskelpoisen ratkaisun, vaikka sitä ei suoritettaisi loppuun asti [8]. Kolmen indeksin ajoneuvovirtausmalli on Fishe-

rin ja Jaikumarin kehittämä muotoilu, mikä määrittelee ajoneuvon kulun kaaren kautta yksilöllisesti kullekin ajoneuvolle. Mallin keskeinen etu on, että se mahdollistaa ajoneuvojen kapasiteetin ja aikarajoitteiden tarkan huomioimisen ilman, että ajoneuvojen on oltava kapasiteetiltaan identtisiä. Malli hyödyntää binäärimuuttujia ajoneuvojen reitin määrittämiseksi ja aikamuuttujia saapumisaikojen hallintaan, mikä tekee siitä erityisen hyödyllisen aikarajoitteita sisältävissä VRP-muunnelmissa. Vaikka malli tarjoaa täsmällisen ratkaisun, sen käytännön soveltaminen saattaa vaatia heuristisia tai likimääräisiä menetelmiä laskennallisen tehokkuuden parantamiseksi.

2.3 Ositusmalli

Moni kapasiteettirajoitettujen ajoneuvojen reititysongelmien täsmällisistä ratkaisumenetelmistä ovat hyviä löytämään optimaalisen ratkaisun tietyissä parametreissa. Kuitenkin oikeassa maailmassa pitää usein huomioida asioiden satunnaisuus. Carlos Novoa et al. esitti artikkelissaan [11] VRP:lle ratkaisumenetelmän, joka huomioi satunnaisuutta. Novoa ja muut hyödyntivät ratkaisumenetelmässä ositusmallia (englanniksi *set-partitioning model* eli *SPM*), joka on hyödyllinen aikarajojen ja monimutkaisten reittirajoitusten tai muiden käytännön rajoitteiden mallintamisessa [11].

Ositusmallin muotoilussa ajoneuvoille valitaan kokonaisia reittejä ennalta määritetystä mahdollisten reittien joukosta sen sijaan, että reittivalinnat tehtäisiin laskemalla kaari kerrallaan, käytetäänkö sitä vai ei. Ositusmallissa määritellään binäärimuuttujat jokaiselle mahdolliselle reitille ja muodostetaan ositusongelma, jolla voidaan löytää optimaalinen CVRP:n rajoitteita noudattava reittikokoelma. Ositusmallissa on etuna myös se, että reittikustannusten ei tarvitse olla suoraan verrannollisia etäisyyteen, vaan se voi ottaa huomioon reittikustannusten epälineaarisuuden toisin kuin kaaripohjaiset mallit [11]. Ositusmalli on hyvin yleispätevä ja voi ottaa huomioon useita reittirajoituksia, koska reitin toteutettavuus sisältyy epäsuorasti reittijoukon R määritelmään [4].

Kuten edellä on todettu, kapasiteettirajoitettujen ajoneuvojen reititysongelmiin usein kuuluu tietyissä oikean maailman tilanteissa myös satunaisuutta. VRP on stokastinen (SVRP), jos yksi tai useampi sen elementeistä mallinnetaan satunnaismuuttujana [11]. Yksi satunnaisista elementeistä voi olla esimerkiksi ajoreitillä olevien asiakkaiden tarpeet. Oikean maailman tilanteita, joissa asiakkaiden tarpeita ei tiedetä on lukuisia ja ne toimivat vastaavasti tilanteissa, joissa tuotteita ei viedä asiakkailta, vaan asiakkailta haetaan jotain. Esimerkiksi jäteautonkuljetuksessa asiakkaiden tarkkaa jätteiden määrää ei tiedetä etukäteen. Seuraavaksi esitellään Carlos Novoan et al. kehittämä yksinkertainen ositusmalli CVRP:lle, jossa asiakkaiden tarpeiden laajuutta ei tiedetä ennen kuin vasta asiakkaan luona. Esitystapa

perustuu lähteeseen [11]. Olkoot

R reittien joukko,
I asiakkaiden joukko,
c_r reitin $r \in R$ tarkka kustannusarvio,
a_{ir} 1, jos asiakas $i \in I$ on reitillä $r \in R$, muussa tapauksessa 0,
k ajoneuvojen määrä,
x_r 1, jos reitti $r \in R$ valitaan, muussa tapauksessa 0.

Jokainen reitti $r \in R$ käy asiakasjoukon luona $A_r \subseteq I$, ja jokaisen ajoneuvon on käytävä kaikilla reittinsä asiakkailla. Reitin r kustannus c_r koostuu kahdesta osasta: reitin kustannuksesta ilman, että reitillä tapahtuu epäonnistumisia, ja kustannuksesta Q_r, joka huomioi varastolle paluusta johtuvan lisämatkan reittivirheiden seurauksena. Reittivirheet ovat tapahtumia, joissa ajoneuvo saapuu asiakkaalle, mutta sillä ei ole tarpeeksi tuotetta tai tilaa täyttämään asiakkaan tarpeita.

Koska jokainen ajoneuvo noudattaa alun perin suunniteltua reittiä, voidaan varastolle paluun odotettu kustannus Q_r laskea yksinkertaisesti etukäteen kullekin reitille. On kuitenkin syytä huomioida, että toisin kuin deterministisen VRP:n tapauksessa, reitin kustannus riippuu reitin suunnasta. Malli voidaan muotoilla minimoimaan odotetut reitityskustannukset seuraavasti

$$\min \quad \sum_{r \in R} c_r x_r \tag{16}$$

$$\text{s.t.} \quad \sum_{r \in R} a_{ir} x_r = 1, \quad \forall i \in I \tag{17}$$

$$\sum_{r \in R} x_r \leq k \tag{18}$$

$$x_r \in \{0, 1\}, \quad \forall r \in R \tag{19}$$

Kohdefunktio (16) minimoi odotetut reittikustannukset. Rajoite (17) varmistaa, että jokaista asiakasta palvellaan täsmälleen yhdellä reitillä. Rajoite (18) takaa, että valittujen reittien määrä ei ylitä ajoneuvojen määrää. Rajoite (19) on binäärimuuttuja. Jokainen mahdollinen reitti joko valitaan osaksi ratkaisua (1) tai ei valita (0).

Reitin kustannusarvioon c_r sisältyy kustannus Q_r, joka huomioi ylimääräisen varastolle suuntautuvan matkan, mikäli reitillä tapahtuu reittivirhe. Tämä kustannus Q_r riippuu reitin suunnasta. Kustannus Q_r voidaan laskea reitille r menettämättä yleispätevyyttä määrittämällä reitti suuntajärjestyksessä seuraavasti $r^{\rightarrow} = \{0, 1, 2, \ldots, s_r, 0\}$, missä 0 edustaa varastoa, 1 ensimmäistä asiakasta reitillä ja niin edelleen. Ongelman kuvauksesta muistetaan, että d_i edustaa asiakkaan i todellista kysyntää ja Q ajoneuvon kapasiteettia. Olkoon $d_{i,j}$ asiakkaiden i ja j välinen etäisyys.

Kaava (20) määrittää tarkasti reitin r lisämatkojen odotetun kustannuksen varastolle annetussa suunnassa $r^{\rightarrow}$. Se on lisäksi sovellettavissa riippumatta siitä ku-

vataanko kysyntä jatkuvina vai diskreetteinä arvoina.

$$Q_{r\rightarrow} \overset{\text{def}}{=} 2\sum_{j=1}^{s_r}\sum_{l=1}^{\infty}\mathbb{P}\left(\sum_{u=1}^{j-1} d_u < lQ < \sum_{u=1}^{j} d_u\right) d_{0,j} + \sum_{j=1}^{s_r}\sum_{l=1}^{\infty}\mathbb{P}\left(\sum_{u=1}^{j-1} d_u = Q\right)(d_{0,j-1} + d_{0,j} - d_{j-1,j}). \tag{20}$$

Ositusmalli tarjoaa CVRP:n ratkaisemiseksi lähestymistavan, jossa ongelman reitit muodostetaan ja valitaan kokonaisuuksina perinteisen kaaripohjaisen optimoinnin sijaan. Tämä saattaa mahdollistaa tehokkaamman ratkaisun suurille ongelmille, sillä ratkaisutilaa voidaan rajoittaa järkevästi valituilla reittijoukoilla. Vaikka ositusmalli tarjoaa laskennallisesti tehokkaamman lähestymistavan kuin perinteinen kaaripohjainen lineaarinen kokonaislukuohjelmointi, sen käytännön sovellettavuus riippuu reittien valinnan laadusta ja heuristisista menetelmien tehokkuudesta.

3 Heuristinen menetelmä

Ajoneuvojen reititysongelman ratkaisemiseksi on kehitetty kaksi vakioitunutta algoritmiluokkaa. Nämä ovat täsmälliset eli eksaktit algoritmit sekä likimääräiset eli approksimatiiviset algoritmit. Täsmälliset ja likimääräiset algoritmit eroavat siten, että täsmälliset algoritmit ovat hakualgoritmeja, jotka analysoivat kaikki mahdolliset vaihtoehdot, joista se pyrkii löytämään täsmällisen optimaalisen ratkaisun. Tämä lähestymistapa käy läpi koko ratkaisujoukon löytääkseen parhaan mahdollisen ratkaisun, joka esimerkiksi minimoi kustannukset tai maksimoi suorituskyvyn. Likimääräiset algoritmit sen sijaan ovat laskennallisesti tehokkaampia. Kyseiset algoritmit etsivät ratkaisuja, jotka ovat riittävän lähellä optimaalista ratkaisua, jolloin ne eivät vaadi yhtä paljon laskennallista tehokkuutta. Nämä algoritmit tarjoavat hyväksyttäviä ratkaisuja ilman, että kaikkia mahdollisia ratkaisuja täytyy käydä läpi. Likimääräiset algoritmit perustuvat heuristisiin sääntöihin ja älykkäisiin optimointitekniikoihin. Täsmällisten algoritmien vahvuus on niiden kyky löytää paras mahdollinen teoreettinen ratkaisu. Kuitenkin VRP:n korkea laskennallinen monimutkaisuus tarkoittaa, että täsmälliset menetelmät vaativat usein paljon laskenta-aikaa, mikä voi tehdä niistä epäkäytännöllisiä tapauksille, joissa on todella monta mahdollista reittiä. [10]

Likimääräisiä algoritmeja on neljää tyyppiä: heuristiset, metaheuristiset, hyperheuristiset ja matemaattis-heuristiset menetelmät (englanniksi *matheuristics*). Heuristiset algoritmit käyttävät käytännön sääntöjä tai alakohtaista tietoa löytääkseen hyväksyttäviä ratkaisuja nopeasti. Metaheuristiset menetelmät ovat laajemmin käytettyjä heuristisia hakutekniikoita, joita sovelletaan monimutkaisiin optimointiongelmiin. Hyperheuristiset menetelmät puolestaan käyttävät heuristiikoita luodakseen tai valitakseen parempia heuristiikkoja optimointiongelmien ratkaisemiseksi. Matemaattisheuristiset menetelmät yhdistävät heuristisia menetelmiä matemaattiseen mallinnukseen ratkaistakseen optimointiongelmia, kuten CVRP. Näillä algoritmeilla on omat vahvuutensa ja heikkoutensa. Sopivan menetelmän valinta riippuu

ongelman ominaisuuksista, käytettävissä olevasta laskentatehosta ja ratkaisun tavoitellusta tarkkuudesta. [10] Seuraavaksi esitellään Alesianin et al. kehittämä asiakkaiden ryhmittelyyn perustuva heuristinen ratkaisumenetelmä.

Ryhmittele ensin, reititä sitten (englanniksi *Cluster First, Route Second* eli CFRS) on yksi tunnetuimpia heuristisia VRP:n ratkaisumenetelmiä. CFRS:n logiikka perustuu nimensä mukaisesti siihen, että ensin ryhmitellään joukko asiakkaita eri ryppäisiin, minkä jälkeen optimoidaan ryppäiden asiakkaille ajoneuvojen toimitusreitit. Ryhmittelyn tarkoituksena on vähentää tarvittavien muuttujien määrää CVRP:n ratkaisemiseksi. Menetelmän esitystapa perustuu lähteeseen [3].

Jokaista yksittäistä rypästä palvelee vain yksi ajoneuvo ja jokaisessa ryppäässä on vähintään yksi asiakas. Mallilla on seuraavat ominaisuudet

1. Yksi ajoneuvo voi palvella yhtä tai useampaa rypästä,
2. Ryppäässä on oltava vähintään yksi asiakas,
3. Jokainen asiakas kuuluu vain yhteen ryppääseen,
4. Ryppäät määrätään siten, että jokaista ryppään asiakasta voidaan palvella yhdellä ajoneuvolla.

Mallin vaiheet ovat seuraavat:

1. Minimoidaan reittien kustannukset rypäspäihin kapasiteettirajoitetuille ajoneuvoille. Rypäspäät (englanniksi *cluster heads*) edustavat kaikkia ryppään asiakkaita.
2. Jokaiselle ajoneuvolle, joka vierailee yhdessä tai useammassa ryppäässä korvataan rypäspäät asiakkaiden sijainneilla ja ratkaistaan erikseen kauppamatkustajaongelma asiakkaiden välille, jotta saadaan optimaalinen asiakaspalvelujärjestys ryppään sisällä.

3.1 Vaihe 1

Rypäspäät vastaavat ongelmanmäärittelyssä virtuaalisia asiakkaita. Rypäspään kysyntä on sen ryppääseen kuuluvien asiakkaiden kysyntöjen summa. Ensimmäisessä vaiheessa ongelma mallinnetaan eri tavalla kuin perinteisessä CVRP-muotoilussa, jossa huomioidaan yksittäiset asiakkaat rypäspäiden sijaan. Tässä lähestymistavassa asiakkaiden sijainnit korvataan rypäspäiden sijainneilla, jotka edustavat virtuaalisia asiakkaita.

Taulukko 1: Määrittelyt.

U'	$U' = \{1, \ldots, n\}$ on kaikkien asiakkaiden joukko.
q_k	Ryppään k asiakkaiden kokonaiskysyntä.
d_{kl}	Rypäspäiden k ja l välinen etäisyys.
u_i	Asiakkaan i sijainti.
K	Ryppäiden joukko.

R	Maksimihakuetäisyys (suurin sallittu etäisyys asiakkaan ja ryväspään välillä).
W	Maksimimäärä asiakkaita, jotka voivat kuulua yhteen ryppääseen.
V	Käytettävissä olevien ajoneuvojen määrä.
Q	Ajoneuvon kapasiteetti.
μ_i	Asiakkaan i kysyntä.
n	Asiakkaiden määrä.

θ_k	Ajoneuvon kuorma ryppään k palvelemisen jälkeen.
y_{kl}	Binäärimuuttuja, joka osoittaa, onko siirtyminen ryppäästä k ryppääseen l osa ratkaisua.
c_k	Ryväspään k sijainti.
A_k	Asiakasjoukko, joka on liitetty ryppääseen k, $A_k = \{u_i \mid i \in k\}$.
a_k	Asiakasjoukko, joka kuuluu ryppääseen k.

Määrittelemme tässä kahden ryppään c_k ja c_l välisen etäisyyden niiden ryväspäiden sijaintien perusteella, eli $d_{kl} = d(c_k, c_l)$. Tarkemmin ryväspäiden sijaintien määrittely esitellään pseudokoodilla algoritmissa (32). Ensimmäisessä vaiheessa jätämme huomiotta asiakkaiden väliset etäisyydet ryppäiden sisällä. Määrittelemme binäärisen muuttujan y_{kl}, joka saa arvon 1, kun reitti ryppäästä k ryppääseen l on osa ratkaisua. Ryhmitelty CVRP voidaan siis muotoilla yhtälöiden (21)-(29) mukaisesti, mutta on syytä huomata, että muotoilussa minimoidaan reittien kustannukset ryppäiden välille, eli todelliset asiakkaat korvataan virtuaalisilla asiakkailla eli ryväspäillä. CVRP voidaan muotoilla seuraavasti

$$\min_{y_{kl}} \quad \sum_{k\in K}\sum_{l\in K} d_{kl}y_{kl} \tag{21}$$

$$\text{s.t.} \quad \sum_{k\in K} y_{kl} = 1, \quad \forall l \in K \setminus \{0\} \tag{22}$$

$$\sum_{l\in K} y_{kl} = 1, \quad \forall k \in K \setminus \{0\} \tag{23}$$

$$\sum_{k\in K} y_{k0} = \sum_{l\in K} y_{0l} \tag{24}$$

$$\theta_l - \theta_k \geq q_l y_{kl} - Q(1 - y_{kl}), \quad \forall k \in K \setminus \{0\}, l \in K \setminus \{0\}, l \neq k \tag{25}$$

$$q_k \leq \theta_k \leq Q, \quad \forall k \in K \setminus \{0\} \tag{26}$$

$$y_{kl} \in \{0, 1\}, \quad \forall k, l \in K \tag{27}$$

$$\sum_{l\in K} y_{0l} \leq V \tag{28}$$

$$y_{kk} = 0, \quad k \in K. \tag{29}$$

Kohdefunktio (21) etsii minimikustannukset kaikille reiteille, jotka palvelevat kaikkia ryppäitä. Rajoitteet (22) ja (23) ovat sisääntulevien ja ulosmenevien kaarien rajoitteet. Ne varmistavat, että jokaisesta ryppääseen liittyvästä solmusta menee yksi kaari sisään ja yksi kaari ulos. Rajoite (24) katsoo, että varastolta lähtevien ajoneuvojen määrä on sama kuin sinne palaavien ajoneuvojen määrä. Rajoitteet (25) ja (26) ovat kapasiteettirajotteita. Jos $y_{kl} = 0$, rajoite (25) muuttuu muotoon $\theta_l + Q \geq \theta_k$, mikä pätee kaikille $\theta_l, \theta_k \in [0, Q]$. Näin ollen, kun $y_{kl} = 0$, rajoite (25)

ei sido ratkaisua. Toisaalta, kun $y_{kl} = 1$, rajoite (25) asettaa ehdon $\theta_l \geq \theta_k + q_l$. Rajoite (26) varmistaa, että ajoneuvon kuorma lähtiessään ryppäästä k on suurempi tai yhtä suuri kuin ryppään k kokonaiskysyntä eikä ylitä ajoneuvon kapasiteettia Q. Rajoite (27) on standardi binäärimuuttuja, joka saa arvon 1, jos ajoneuvo kulkee reitillä ryppäästä k ryppääseen l. Rajoite (28) varmistaa, että ratkaisussa ei hyödynnetä useampaa ajoneuvoa kuin on määritelty. Rajoite (29) estää ajoneuvoja menemästä ryppäästä k takaisin samaan ryppääseen k. Malli etsii minimikustannusreitit kaikkien ryppäiden palvelemiseksi, mutta ratkaisu ei kuitenkaan etsi minimikustannusreittejä yksittäisille asiakkaille.

3.2 Vaihe 2

Seuraavaksi siirrytään vaiheeseen 2, jossa ratkaistaan kauppamatkustajaongelma jokaiselle ajoneuvolle siten, että rypäspäiden sijainnit korvataan ryppäiden sisällä olevien todellisten asiakkaiden sijainneilla. Jokaisen ajoneuvon TSP:n ratkaisu on lyhin mahdollinen reitti, joka käy kaikilla kyseiselle ajoneuvolle määrätyillä asiakkailla eli kaikilla asiakkailla, jotka kuuluvat sen palvelemiin ryppäisiin. On tärkeää huomata, että kauppamatkustajaongelma ratkaistaan vain niille asiakkaille, jotka on määrätty ajoneuvolle ensimmäisessä vaiheessa. Asiakkaiden palvelujärjestyksen määrää TSP:n ratkaisu, eikä ryppäillä ole enää merkittävää roolia tässä vaiheessa.

Jotta ylempi malli voidaan ratkaista tulee määrittää ryppäät ja niiden rypäspäät. Tämä toteutetaan soveltamalla kaksivaiheista ryhmittelyalgoritmia. Ensimmäisessä vaiheessa asiakkaat osoitetaan alustaviin ryppäisiin. Toisessa vaiheessa alustavia ryppäitä päivitetään, jotta ratkaisu saadaan optimaalisemmaksi. Algoritmi (32) kuvaa algoritmin askeleita käyttäen taulukon (1) määrittelyitä. Rypäspäät määritellään niiden sijaintien c_k perusteella, missä $k = 1, \ldots, |K|$, ja $|K|$ on ryppäiden määrä. On tärkeää huomata, että $|K|$ voi muuttua iteraatioiden välillä, jos osaa asiakkaista ei voi määrätä ryppääseen tai jokin rypäs jää tyhjilleen.

Rypäspäät ovat keskiöitä, jotka edustavat tiettyä asiakasjoukkoa. Algoritmi (32) aloitetaan valitsemalla $|K|$ satunnaista rypäspäätä. Seuraavaksi määrätään asiakkaat asiakkaiden sijaintien perusteella niiden lähimmälle rypäspäälle, mikäli tämä ei riko yhtälöiden (21)-(29) rajoitteita. Olkoon $0 < |K| \leq V \leq n$, missä V on käytettävissä olevien ajoneuvojen kokonaismäärä ja n on asiakkaiden lukumäärä. Olkoon c_k ryppään $k \in K$ rypäspään sijainti. Määritellään myös a_k joukoksi asiakkaita, jotka kuuluvat kyseiseen ryppääseen $a_k = \{i \mid i \in k\}, i \in \{1, \ldots, n\}$, jossa i on yksittäinen asiakas. Olkoon A_k joukko asiakassijainteja, jotka liittyvät kyseiseen ryppääseen. Jokaisella asiakkaalla $i \in \{1, \ldots, n\}$ on asiakassijainti u_i, joten $A_k = \{u_i \mid i \in k\}$. Määritettyjen asiakkaiden joukko on $a = \bigcup_k a_k$, kun taas määrittämättömien asiakkaiden joukko on $U = U' \setminus a$, ja kaikkien asiakkaiden joukko $U' = \{1, \ldots, n\}$. Vastaavasti, olkoon $C = \{c_1, \ldots, c_{|K|}\}$ joukko rypäspäiden sijainteja. Aluksi jokaisen ryppään k rypäspään sijainti c_k alustetaan valitsemalla satunnaisesti yksi asiakaspaikoista $(u_1, \ldots, u_n)$ siten, että jokainen asiakaspaikka on yhtä todennäköinen valinta. Kaksi eri rypäspäätä eivät saa saada samaa sijaintia. Toisin sanoen $c_k \sim \mathcal{U}(u_1, \ldots, u_n)$ siten, että $c_k \neq c_j$, $j \in K$. Aluksi oletetaan, että kaikki ryppäät ovat tyhjiä, eli $a_k = \emptyset, k \in K$. Kun rypäspäiden sijainnit c_k, $k \in K$ on satunnaisesti valittu, suoritetaan seuraavat vaiheet. Vaiheessa 1 järjestetään asiakkaat

$i \in \{1, \ldots, n\}$ etäisyyden perusteella niiden lähimpään ryväspäähän. Tämä varmistaa, että ryppäisiin priorisoidaan asiakkaita, jotka ovat lähimpänä ryväspäitä. Jos asiakasta ei voida lisätä lähimpään ryppääseen kapasiteettirajoitusten takia, tiedetään, että kaikki ryppääseen aikaisemmin sijoitetut asiakkaat ovat kyseistä asiakasta rypästä lähempänä. Asiakkaan $i \in \{1, \ldots, n\}$ etäisyys lähimpään ryväspäähän $k \in K$ on

$$D_i = \min_k d_{i,k},$$

missä $d_{i,k}$ on asiakkaan i ja ryväspään c_k välinen etäisyys. Kun asiakkaiden etäisyydet niiden lähimpään ryväspäähän $D_1, D_2, \ldots, D_n$ on laskettu, kartoitetaan asiakkaat $(1, 2, \ldots, n)$ prioriteettijoukkoon $P = (b_1, b_2, \ldots, b_n)$, jossa asiakasprioriteetit täyttävät ehdon $D_{b_1} \leq D_{b_2} \leq \cdots \leq D_{b_n}$. Tämä vaihe voidaan toteuttaa laskennallisesti algoritmin (30) pseudokoodin mukaisesti.

$$\begin{aligned}
&\text{1: syöte:} \quad \{d_{i,k}\}, U' \\
&\text{2: jokaiselle } i \in U' \text{ laske } D_i \text{ ratkaisemalla } \min_k d_{i,k} \\
&\text{3: aseta:} \quad P = (b_1, \ldots, b_n) \text{ siten, että } D_{b_1} \leq \cdots \leq D_{b_n} \\
&\text{4: tulos:} \quad P = (b_1, \ldots, b_n).
\end{aligned} \tag{30}$$

Vaiheessa kaksi määrätään asiakkaat ryppäisiin. Aloitetaan prioriteettijonon ensimmäisestä asiakkaasta b_1 ja suoritetaan seuraavat vaiheet. Ensin alustetaan joukko $C_{b_1} = K$, joka sisältää kaikki mahdolliset ryppäät, joihin asiakas b_1 voidaan määrätä. Tämän jälkeen määritetään asiakkaan b_1 lähin ryväs k^* seuraavasti

$$k^* = \arg\min_k d_{b_1,k},$$

jonka jälkeen tarkistetaan seuraavat ehdot:

- Ryväs k^* ei ole saavuttanut asiakkaidensa maksimimäärää W.
- Etäisyys d_{b_1,k^*} on pienempi tai yhtä suuri kuin suurin sallittu etäisyys asiakkaan ja ryväspään R välillä.
- Kyseisen ryppään kokonaisasiakaskysyntä $\mu_{b_1} + \sum_{j \in a_k^*} \mu_j$, missä $\sum_{j \in a_k^*} \mu_j$ on niiden asiakkaiden kokonaiskysyntä, jotka ovat jo ryppäässä k^*, on pienempi tai yhtä suuri kuin ajoneuvon kapasiteetti Q.

Jos kaikki yllä olevat ehdot täyttyvät, lisätään asiakas b_1 ryppään k^* asiakkaiden joukkoon a_k^*. Jos jokin ehto ei täyty, asiakasta b_1 ei lisätä ryppääseen k^* ja kyseinen ryväs poistetaan mahdollisten ryppäiden joukosta asiakkaalle b_1: $C_{b_1} \leftarrow C_{b_1} \setminus \{k^*\}$. Tämän jälkeen suoritetaan samat tarkistukset muille ryppäille joukossa C_{b_1}, kunnes asiakas b_1 voidaan lisätä johonkin ryppääseen. Jos asiakas b_1 lisätään ryppääseen k, päivitetään kyseisen ryppään keskipiste, joka edustaa ryppään kaikkien asiakkaiden painopistettä. Laajennetun joukon a_k ryväspään sijainti c_k lasketaan seuraavasti

$$c_k = \frac{1}{|a_k|} \sum_{i \in a_k} u_i,$$

jossa $|a_k|$ on niiden ryppääseen k kuuluvien asiakkaiden lukumäärä, jotka ovat tallennettu joukkoon a_k. Tämä vaihe päättyy, kun kaikki prioriteettijonossa olevat asiakkaat on käsitelty. Toinen vaihe voidaan toteuttaa laskennallisesti algoritmin (31) pseudokoodin mukaisesti.

$$
\begin{aligned}
&\text{1: syöte:} \quad \{u_i\}, W, R, \{\mu_i\}, K, \{c_k\}, \{d_{i,k}\}, U \\
&\text{2: laske P suorittamalla algoritmi (30)} \\
&\text{3: jokaiselle } b_i \in P \text{ aseta } C_{b_i} = K \\
&\text{4: kun } b_i \in U \quad ja \quad C_{b_i} \neq \emptyset \text{ aseta } k^* = \arg\min_k d_{b_i,k} \\
&\text{5: jos } |a_{k^*}| < W \quad ja \quad d_{b_i,k^*} \leq R \quad ja \quad \mu_{b_i} + \sum_{j \in a_{k^*}} \mu_j \leq Q \text{ niin} \\
&\text{6: } a_{k^*} \leftarrow a_{k^*} \cup \{b_i\} \\
&\text{7: } c_{k^*} := \frac{1}{|a_{k^*}|} \sum_{i \in a_{k^*}} u_i \\
&\text{8: } U = U \setminus \{b_i\} \\
&\text{9: muuten} \\
&\text{10: } C_{b_i} \leftarrow C_{b_i} \setminus \{k^*\} \\
&\text{11: tulos:} \quad \{a_k\}, \{c_k\}
\end{aligned}
\tag{31}
$$

Kolmannessa vaiheessa poistetaan kaikki asiakkaat, jotka on osoitettu rypäsjoukkoihin a_k, ja säilytetään vain päivitetyt rypäspäiden sijainnit c_k, jossa $k \in K$. Näillä päivitettyjen rypäspäiden sijainneilla suoritetaan vaiheita 1 ja 2, kunnes asiakkaiden määrääminen ryppäisiin kahdessa peräkkäisessä algoritmin iteraatiossa ei muutu eli algoritmi on konvergoitunut. Jos algoritmi on konvergoitunut mutta kaikkia asiakkaita ei ole osoitettu johonkin rypäsjoukon K ryppääseen tai joissakin joukon ryppäissä ei ole yhtään asiakasta, kasvatetaan tai vähennetään ryppäiden lukumäärää $|K|$, jonka jälkeen suoritetaan uudelleen kaikki ryhmittelyalgoritmin vaiheet. Tämä algoritmin itseohjautuvuus takaa, että kaikki asiakkaat määrätään ryppäisiin. Vaiheen kolme voi toteuttaa laskennallisesti algoritmin (32) pseudokoodin mukaisesti.

$$
\begin{aligned}
&\text{1: kun } a_k \neq \emptyset, \quad k \in K \wedge U \neq \emptyset \text{ tee} \\
&\text{2: } c_k \sim \mathcal{U}(u_1, \ldots, u_n), \quad k \in K, \text{ siten että } c_k \neq c_j, \quad j \in K \\
&\text{3: } U \leftarrow \{1, \ldots, n\} \\
&\text{4: } a_k \leftarrow \emptyset, \quad k \in K \\
&\text{5: kun } \left(\{a_k\}_1^{|K|}\right)_{\text{edellinen}} \neq \left(\{a_k\}_1^{|K|}\right) \text{ tee} \\
&\text{6: } \left(\{a_k\}_1^{|K|}\right)_{\text{edellinen}} \leftarrow \left(\{a_k\}_1^{|K|}\right) \\
&\text{7: } \left(\{a_k\}_1^{|K|}\right) \leftarrow \text{algoritmi (31)} \\
&\text{8: jos } U \neq \emptyset, \text{ niin} \\
&\text{9: } |K| \leftarrow |K| + 1, \text{ jos on osoittamattomia asiakkaita} \\
&\text{10: jos } \exists k, a_k = \emptyset, \text{ niin}
\end{aligned}
\tag{32}
$$

11: $|K| \leftarrow |K| - 1$, jos on ryppäitä ilman asiakkaita: $a_k = \emptyset$, jollekin $k \in K$
12: tulos: $\{c_k\}, \{a_k\}$

On syytä huomata, että asiakkaiden määrääminen ryppäisiin, rypäspäiden sijaintien päivitys ja lopetusehto perustuvat löyhästi k-means-algoritmiin, joka esitellään opetusmateriaalissa [12]. Yksi merkittävä ero algoritmien välillä on, että asiakkaiden määräysvaiheessa asiakasta ei aina sijoiteta lähimpään rypäskeskukseen, koska lisäksi on otettava huomioon useita etäisyyteen ja kapasiteettiin liittyviä rajoituksia. Lisäksi ryppäiden lukumäärä $|K|$ voi muuttua, jos algoritmi ei onnistu kohdistamaan kaikkia asiakkaita ryppäisiin.

3.3 Tulokset

Alesianin et al. menetelmä muotoilee alkuperäisen CVRP-ongelman pienemmän ulottuvuuden ongelmaksi (englanniksi *reduced dimension problem*). Vaikka menetelmän aikavaativuuden parannus on eksponentiaalinen, etsitty ratkaisujoukko pienenee, mikä saattaa johtaa optimaalisen ratkaisun poissulkemiseen. Menetelmän kehittäjät vertailivat mallin tehokkuutta arvioimalla laskennallisia kustannuksia ja ratkaisun optimaalisuuden eroa käyttämällä julkisesti saatavilla olevia testitapauksia, joita käytetään CVRP-ratkaisumenetelmien vertailuun. Optimaalisuuden ero on menetelmän ratkaisun ero optimaalisesta ratkaisusta. Laatueroa mitataan suhteellisena eroavaisuutena seuraavien ratkaisujen välillä:

- Esitellyn menetelmän ratkaisu.
- Paras tunnettu tai optimaaliseksi todistettu ratkaisu, joka on raportoitu osoitteessa http://vrp.galgos.inf.puc-rio.br.

On syytä huomata, että menetelmän ratkaisemiseksi on käytettävä branch-and-cut-menetelmää laskemaan globaali optimaalinen ratkaisu tai heuristista lähestymistapaa laskemaan likimääräinen ratkaisu. Lisäksi jokaiselle ajoneuvolle täytyy ratkaista TSP-ongelma, mikä voi johtaa suurempiin laskennallisiin kustannuksiin tietyissä tapauksissa. Tätä tutkitaan laskennallisissa kokeissa, joissa arvioidaan menetelmän laskennallisia kustannuksia ja ratkaisun laatua. TSP-ongelmat ratkaistaan tspy-kirjaston TwoOpt-algoritmilla Pythonilla.

TwoOpt -algoritmin kehitti G. A. Croes, ja se esiteltiin hänen vuonna 1958 julkaistussa teoksessa *"A Method for Solving Traveling-Salesman Problems"*. Tämä paikallisen haun algoritmi on suunniteltu parantamaan TSP:n reittejä. Se tunnistaa kohdat, joissa reitti risteää itsensä kanssa, ja järjestää solmut uudelleen siten, että risteämä poistuu. Algoritmissa kokeillaan kaikkia kaaripareja ja tarkistetaan, onko risteämätön versio lyhyempi kuin alkuperäinen. [13]

Kaikki testit suoritetaan yleiskäyttöisellä tietokoneella, jossa on 2,3 GHz Intel Core i5 -prosessori ja 16 Gt RAM-muistia. Testitapaukset sisältävät seuraavat tapauskoot, jotka ovat julkisesti saatavilla osoitteessa http://vrp.galgos.inf.puc-rio.br:

- Pienet tapauskoot: 32–82 asiakasta.
- Keskikokoiset tapauskoot: 101–350 asiakasta

- Suuret tapauskoot: 350–1001 asiakasta.
- Hyvin suuret tapauskoot: 3000–16000 asiakasta.

Kokeessa raportoidaan suorituskyky, kun ylempi malli ja alkuperäinen CVRP-malli ratkaistaan 100 sekunnin aikarajan sisällä. Tämän jälkeen verrataan ylemmän mallin ja alkuperäisen CVRP:n ratkaisua testitapauksen optimaalisimpaan tunnettuun ratkaisuun. Analyysin tavoitteena on verrata optimaalisuutta eli ylemmän mallin ratkaisun eroa yleiseen optimaaliseen ratkaisuun ja suorituskykyä tutkimalla kuinka paljon paremmin malli suoriutuu alkuperäiseen CVRP:hen verrattuna, kun käytetään 100 sekunnin laskenta-aikaa. Jälkimmäinen analyysi osoittaa, voiko ryhmittely parantaa ratkaisuja rajoittamalla haettavaa ratkaisutilaa ja siten nopeuttaa optimoinnin prosessia. Malli saavutti

- pienissä tapauskoissa keskimäärin $10,1\%$ optimaalisuuseron $4,8\%$ keskihajonnalla,
- keskikokoisissa tapauskoissa keskimäärin 9% optimaalisuuseron $4,4\%$ keskihajonnalla,
- suurissa tapauskoissa keskimäärin 8.9% optimaalisuuseron 4% keskihajonnalla ja
- hyvin suurissa tapauskoissa keskimäärin $15,7\%$ optimaalisuuseron 5.3% keskihajonnalla.

Alkuperäinen CVRP-malli saavutti pienissä tapauskoissa keskimäärin $11,4\%$ optimaalisuuseron. Tämä 1.3% korkeampi tulos osoittaa, että esitelty menetelmä hyödynsi 100 sekunnin laskenta-aikabudjetin tehokkaammin löytääkseen parempia ratkaisuja. Tarkemmat laskennalliset analyysit löytyvät lähteestä [3] ja sivustolta http://vrp.galgos.inf.puc-rio.br., jossa testitapaukset ja niiden yleiset optimaaliset ratkaisut ovat saatavilla.

4 Yhteenveto

Kapasiteettirajoitettujen ajoneuvojen reititysongelma on laajennus yleisestä ajoneuvojen reititysongelmasta ja yksi maailman tutkituimmista matemaattisen optimoinnin ongelmista. Kapasiteettirajoitus tarkoittaa mitä vain kuormallista rajoitetta. Näitä voivat olla esimerkiksi tilataksin matkustajamäärä, toimitettavien pakettien määrä, pakettien kokonaispaino tai kokonaistilavuus. Kapasiteettirajoitus voi koskea toimitusten enimmäiskuorman lisäksi myös sitä, kuinka paljon tavaraa asiakkailta voi kerralla noutaa. CVRP:n kapasiteettirajoitusten lisäksi oikeassa maailmassa rajoituksia on useasti enemmän. Näitä voivat olla muun muassa käytettävien ajoneuvojen määrä, ajoneuvojen reittien maksimipituus tai aikaikkunarajoitus. Eri ongelmatilanteille on eri ratkaisumenetelmiä, joilla on omat vahvuutensa ja heikkoutensa. Sopivan menetelmän valinta riippuu ongelman ominaisuuksista, käytettävissä olevasta laskentatehosta ja ratkaisun tavoitellusta tarkkuudesta.

Tutkielmassa esiteltiin yleisesti CVRP:n historia, mallinnus ja epätäsmälliset ratkaisumenetelmät. Siinä käsiteltiin kolmea eri mallinnustapaa: lineaarinen kokonaislukuohjelmointimalli, kolmen indeksin ajoneuvovirtausmalli sekä ositusmalli. CVRP kuuluu NP-vaikeisiin optimointiongelmiin. Sille ei tunneta muita kuin expontentiaalisen ajan vieviä algoritmeja, jotka ratkaisisivat kaikki mahdolliset CVRP-tapaukset optimaalisesti. Tämän takia CVRP:n ratkaisemiseksi on kehitetty useita epätäsmällisiä ratkaisumenetelmiä. Ratkaisumenetelmät ovat epätäsmällisiä, koska ne eivät käy kaikkia mahdollisia reittejä läpi vaan etsivät ratkaisun, joka on riittävän lähellä optimaalista ratkaisua. Ne ovat hyödyllisiä oikean maailman tilanteissa, joissa tietokoneen laskentateho on rajoitettu.

Tutkielmassa esiteltiin asiakkaiden ryhmittelyyn perustuva epätäsmällinen mutta laskennallisesti tehokas heuristinen menetelmä. Menetelmä on Francesco Alesianin et al. kehittämä. Se perustuu yhteen tutkituimmista CVRP:n heuristisista menetelmistä: *Ryhmittele ensin, reititä sitten* (CFRS). Ryhmittelyn tarkoituksena on vähentää tarvittavien muuttujien määrää CVRP:n ratkaisemiseksi. Menetelmässä asiakkaat jaetaan ensin ryppäisiin, jonka jälkeen ryppäiden välille optimoidaan ajoneuvoille reitit siten, että yhtä rypästä palvellaan yhdellä ajoneuvolla, mutta yksi ajoneuvo voi palvella useampaa kuin yhtä rypästä. Mallin ensimmäisessä vaiheessa minimoidaan reittien kustannukset ajoneuvoille rypäspäihin, jotka edustavat kaikkia ryppään asiakkaita. Toisessa vaiheessa jokaiselle ajoneuvolle korvataan rypäspäät asiakkaiden sijainneilla ja ratkaistaan erikseen kauppamatkustajaongelma asiakkaiden välille, jotta saadaan optimaalinen asiakaspalvelujärjestys ryppään sisällä. Menetelmän kehittäjät vertasivat menetelmäänsä suorituskyvyltä ja optimaalisuudelta alkuperäiseen CVRP-malliin arvioimalla niiden laskennallisia kustannuksia. Muunneltu ryhmittelymalli osoittautui laskennallisesti tehokkaammaksi ainakin pienissä tapauskoissa, joissa on 32–82 asiakaspistettä.

Viitteet

[1] AIMMS. Capacitated vehicle routing problem formulation, 2024. Accessed: February 5, 2025.

[2] AIMMS. Explicit dantzig-fulkerson-johnson formulation, 2024. Accessed: February 5, 2025.

[3] F. Alesiani, G. Ermis, , and K. Gkiotsalitis. Constrained clustering for the capacitated vehicle routing problem. *Journal of Experimental & Theoretical Artificial Intelligence*, 34(6):679–695, 2022.

[4] R. Baldacci, E. Bartolini, and A. Mingozzi. An exact algorithm for the vehicle routing problem with capacitated vehicles. *Mathematical Programming*, 113(2):381–410, 2008.

[5] G. B. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Operations Research*, 2(4):393–410, 1954.

[6] Encyclopædia Britannica. Nonlinear programming, 2024. Accessed: February 4, 2025.

[7] H. Jiang, M. Lu, Y. Tian, J. Qiu, and X. Zhang. An evolutionary algorithm for solving capacitated vehicle routing problems by using local information. *Applied Soft Computing*, 117:108431, 2022.

[8] G. Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3):345–358, 1992.

[9] G. Laporte, P. Toth, and D. Vigo. Vehicle routing: Historical perspective and recent contributions. *EURO Journal on Transportation and Logistics*, 2, 05 2013.

[10] D. Muriyatmoko, A. Djunaidy, and A. Muklason. Heuristics and metaheuristics for solving capacitated vehicle routing problem: An algorithm comparison. *Procedia Computer Science*, 234:494–501, 2024.

[11] C. Novoa, J. Linderoth, et al. A set partitioning-based model for the vehicle routing problem with stochastic demands. Technical report, University of Wisconsin-Madison, 2006.

[12] C. Piech. K-means clustering, 2013. Accessed: February 19, 2025.

[13] J. J. A. Slootbeek. Average-case analysis of the 2-opt heuristic for the tsp. Master's thesis, University of Twente, 2017.

[14] H. Yarar. What is vehicle routing problem?, 2024. Accessed: February 4, 2025.